## REMARKS

Examiner Pollock is thanked for his thorough review of the pending application.

The drawing informalities have been addressed by the submission of replacement pages, which are legible and that delete extraneous material. Entry of these replacement pages is respectfully requested.

As requested, the abstract has been amended and is now less than 150 words.

The Examiner's rejection of claims 1 and 11 as being indefinite has been noted, and each of these claims has been reworded and should now be clear. In particular, as positively recited, the "inhibiting" function takes place at the content server that has already executed the purge of a "given content file." After that "given content file" has been purged, the given content server is inhibited from receiving that given content file (the same one it has already purged) from another content server that has not yet performed the purge of that file. This operation is quite useful in the context of a content delivery network wherein a set of content servers is sharing content files. This advantage can be explained by considering what would happen if this feature were not employed. Thus, e.g., assume that content server A and content server B are sharing content files and that each of these servers has the capability to request a given content file C from the other in the event the requesting server receives an end user request for the given content file C but does not have that file for whatever reason. Now, assume content server A and content server B have each been instructed to purge the given content file C, but that these servers carry out the purge independently. In such case, content server A may have purged given content file C while content server B has not yet done so. If, in this example, content server A then receives a new end user request for file C, it will check its local storage and determine that the file is not present; accordingly, content server A may then request file C from content server B with whom it is sharing content files. In this situation, it is desirable to "inhibit" content server A from again receiving file C, which is just purged.

This function is now positively recited in claim 1. In particular, "after a first content server in the set of content servers has executed the aggregate purge request and, as a result, has purged a given content file," the claim further requires "inhibiting the first content server from receiving the given content file from a second content server in the set of content servers with which the first content server shares content files if the second content server has not then

executed the aggregate purge request." Claim 11 describes this function is a similar manner, namely "with respect to a given content file that has been removed from the given content server, ... inhibiting the given content server from receiving the given content file from at least one other content server with which the given content server shares content files if the other content server has not then removed the given content file that has been removed by the given content server." A mechanism that facilitates this function is described on pages 31-33 of the written description and involves the use of "sequence numbers" that are maintained and passed between the content servers that share files.

The Examiner states that "it is unclear whether the first server enters negotiation with one server, whether the second server's functionality is inhibited, whether the second server may send other files" and "what mechanism is used to inhibit this sharing and whether that mechanism operates on the first server, the second server, or both." Respectfully, the Examiner's comments in this regard are noted but they do not go to the question of whether the previously-presented claim language was indefinite. By these comments, the Examiner appears to be suggesting that additional limitations be included in the claim language, but this is deemed unnecessary as the functional language is all that is required to describe this feature of the invention. As to the "mechanism," as noted at least one mechanism is described on pages 31-33 of the written description and involves a "sequence number" technique. The prior art, however, does not dictate that this specific mechanism be claimed. Also, it is not seen why the claim needs to describe whether the servers enter into negotiation with one another, and the claim language (which focuses on the "first" server) need not say anything about the "second server's functionality." Any such limitations would be merely superfluous and would unnecessarily restrict the claim in a manner that is not required by the prior art.

Each of claims 1 and 11 now particularly point out and distinctly claim the subject matter, and withdrawal of the indefiniteness rejection thus is respectfully requested.

Claims 1 and 11 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Ofek in view of Schmuck et al. This rejection is respectfully traversed.

With respect, the Examiner has ignored numerous meaningful limitations in the claims in applying the cited references. In the first instance, both Ofek and Schmuck et al. simply concern file management in distributed file systems. Ofek, the primary reference, relates to computer-implemented "database management systems" and the problem of maintaining data integrity

across a set of mirrored machines. In particular, the inventor there described his invention as a "data processing system that stores a data base on redundant storage devices and that enables applications, such as decision support system applications, to run concurrently with other applications, such as on-line transaction processing applications." The present invention, in contrast, concerns (in claim 1) "identifying and removing given content files from a set of content servers in a content delivery network." In like manner, claim 11 relates to an improvement to "a content delivery network wherein third party content is cached on and served from a set of content servers in response to end user requests." A "content delivery network" is described generally on pages 1-2 of the written description; it is typically operated by a service provider on behalf of participating content providers who use the CDN content servers to cache and serve web page content, streaming media and software downloads. A CDN is unrelated to a "decision support system application" or a "database management" system, which is the described operating environment in Ofek. Moreover, in a CDN, it is typically not the case that the servers host the same content or (in the words of Ofek) are "mirrors" of one another. Thus, in the first instance one of ordinary skill in the art would not even consider the distributed file systems described in Ofek or Schmuck et al. as analogous art.

Moreover, in applying Ofek, the Examiner has actually rewritten several claim limitations, which is improper. Thus, for example, the rejection in paragraph 16a refers to "identifying content files to be modified within the content servers" and in paragraph 16d to "writing to the content server each content file identified in the aggregate write request." The Examiner here appears to be equating a file write operation with the CDN file purge request, which rewrites what the applicants are claiming. Obviously, it is not appropriate to rewrite the claim language and then reject the rewritten language using a reference that does not describe or suggest what is actually claimed. That is the case here. Ofek does not concern purging content files from CDN content servers; the reference says nothing about purging or file removal, let alone "pushing an aggregate purge request to each of set of staging servers" or "periodically, having each of [a] set of content servers obtain the aggregate purge request ..." Indeed, this hierarchical "push-pull" aspect of the invention, e.g., use of a set of staging servers that receive aggregate purge requests pushed to them, where a given content server then obtains the request (in effect, pulls it) from a staging server, is nowhere disclosed or suggested in Ofek. Rather, the Examiner's citation (steps 62 and 65) concerns merely writing data to a memory location and

then subsequently sending some signal to a host system. How these steps map to the claimed steps is totally unclear (they do not). The Office bears the burden of applying the cited art in a clear and unequivocal manner. With all due respect, the Examiner has not shown how the claimed steps (which are quite specific) read on Figure 3 steps 62 and 65 or any associated text.

In particular, the cited steps 62 and 65 say nothing about staging servers or aggregate purge requests, or anything remotely relevant thereto. The claim steps explicitly concern how to create an aggregate purge request (not simply a file write request) and how that request is then provided to a set of distributed content servers in a CDN using, for example, a "push-pull" type of technique. Stated another way, the "writing" of data to the system memory (step 62) and then sending some signal to a host (step 65) is not a "push-pull" concept as is positively recited in the claim. Further, the claim also requires that the "content servers obtain the aggregate purge request independently and at different times." The Examiner assumes this to occur (see paragraph 16c), but he does not offer any supporting citation in the reference itself. The feature is not present in Ofek, which says nothing about having content servers obtain an aggregate purge request to remove a set of content files therefrom.

In paragraph 17, the Examiner errs in concluding, without evidence or foundation, that "one of ordinary skill in the art would have noted that a delete file falls under the definition of a write file command." In particular, conclusory statements regarding alleged common knowledge are not evidence that can support an obviousness rejection. In Re Sang Su Lee, No. 00-1158 (Fed. Cir. 2002).

In paragraph 18, the Examiner argues that Ofek's restore command meets the inhibiting function now positively recited in the claims. This is a misreading. In Ofek, the RESTORE command restores all the data on a given device to the devices that are mirrored thereto. As the patent states, this "procedure is useful if a failure occurs in the ... mirror devices ... while the [given] device ... has a valid copy." Restoring (i.e. writing) data to a given mirror device is exactly the opposite of what is claimed, namely, inhibiting (e.g., preventing the writing to) a first content server from receiving another copy of a content file that it recently purged. The Examiner's strained interpretation is incorrect from a technical perspective and is a hindsight reconstruction based on the wording of the claim limitation, not the actual teachings of the reference.

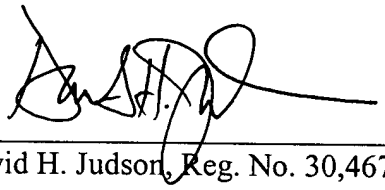
Schmuck et al. do not make up for the deficiencies in the primary reference. It does not concern purging content files from a content delivery network, the push-pull mechanism described above, or any technique to inhibit a particular content server from re-obtaining a content file that it recently purged. At most, Schmuck et al. simply discuss deleting files in a distributed file system.

Stated simply, these references do not disclose or suggest any of the specific limitations set forth in each independent claim. They concern different types of systems and address technical matters that are not being claimed here.

Claims 1 and 11 describe patent subject matter. Claims 2-8 are patentable for the reasons advanced with respect to claim 1.

A Notice of Allowance is respectfully requested.

Respectfully submitted,

By: 

David H. Judson, Reg. No. 30,467

ORIGINAL DRAWINGS – ANNOTATED TO SHOW CHANGES

Annotated Sheet




Redrawn

Figure 1

Annotated Sheet

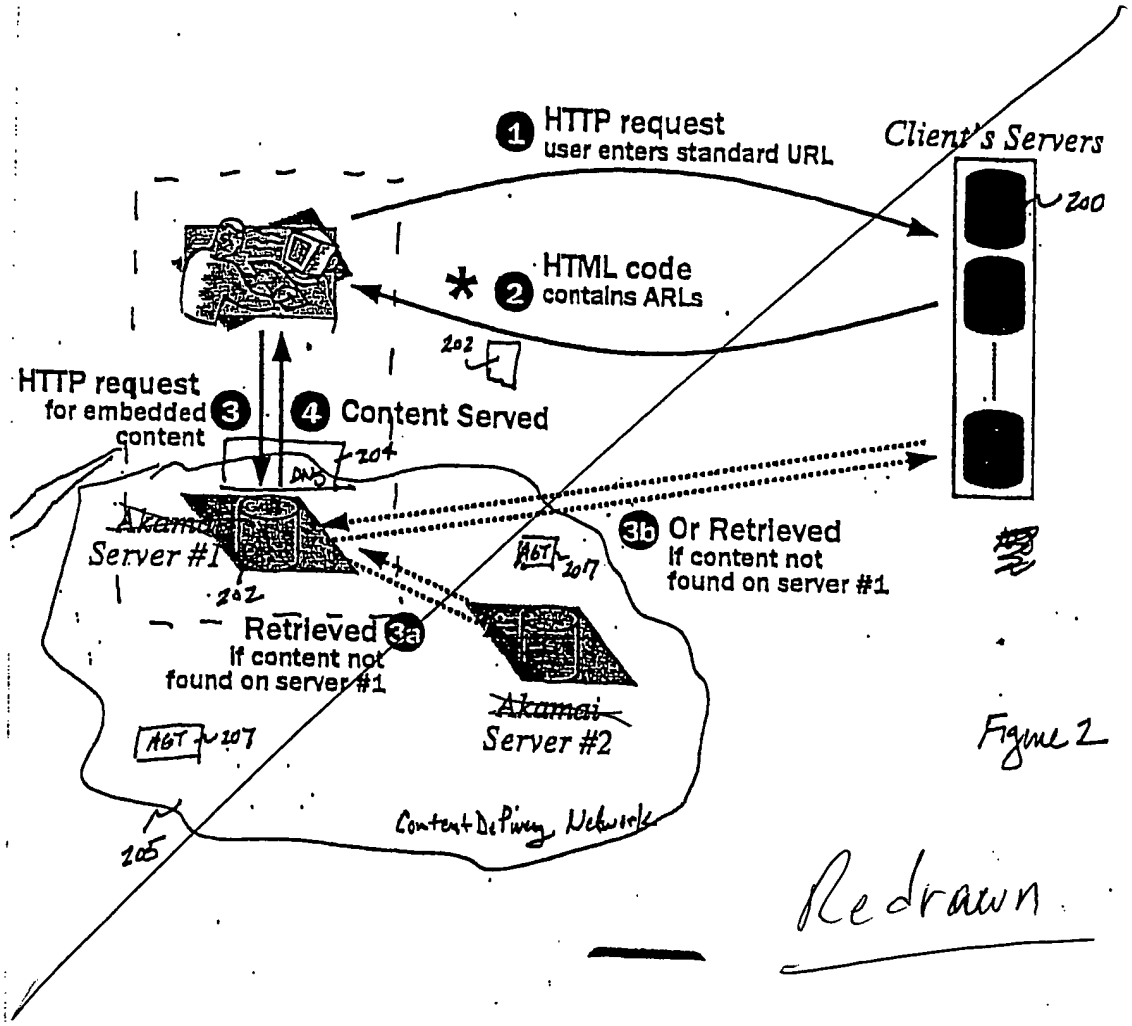
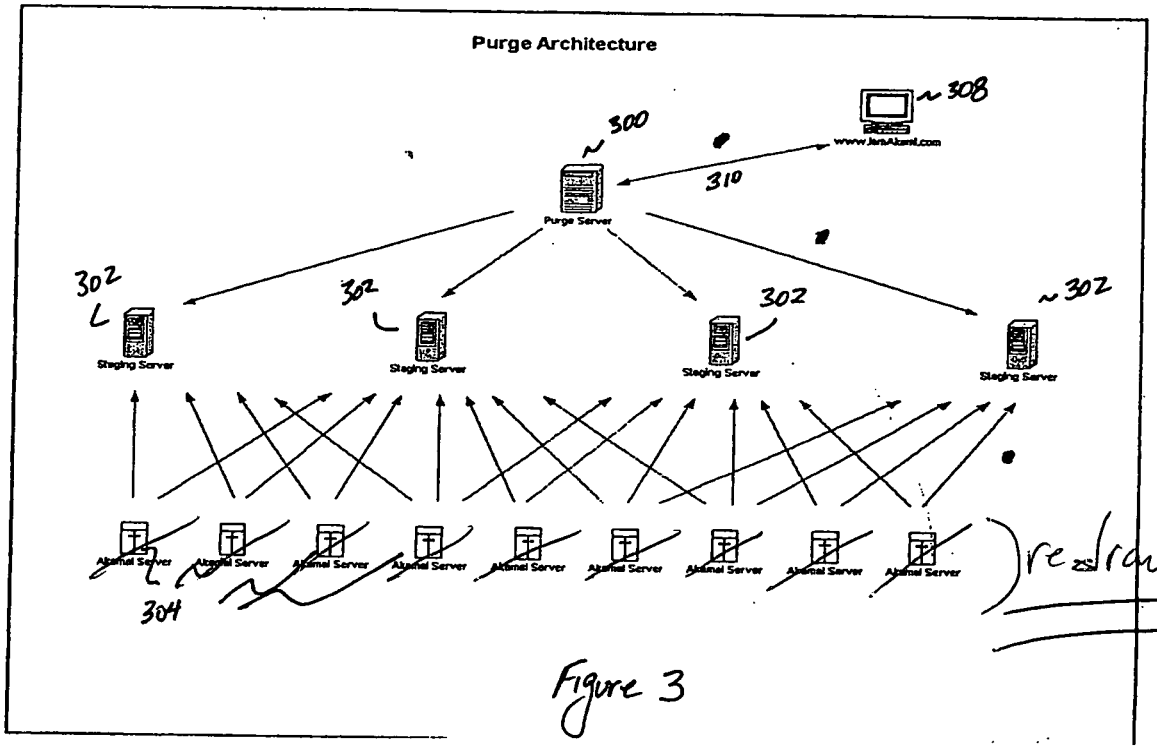



Figure 2

Redrawn




redrawn

Figure 3

3/6

ABC ABC

*ARL purge results for Akamai Technologies*

Your purge request has been submitted to the Akamai system.
Your purge request id code is: **782**. Please record this number for future reference.
The estimated maximum completion time for this request is: **480 secs.**
The following ARLs have been submitted:

| # | ARL | Purge status |
|---|---|---|
| 1 | http://f/4/550/4d/build.akamai.com/cgi-bin/test.pl | Submitted |

Figure 6

moved to separate Sheet

Annotated Sheet

416

Annotated Sheet

@ Purge App | @ Purge Server | @ Staging Server | @ Content Server

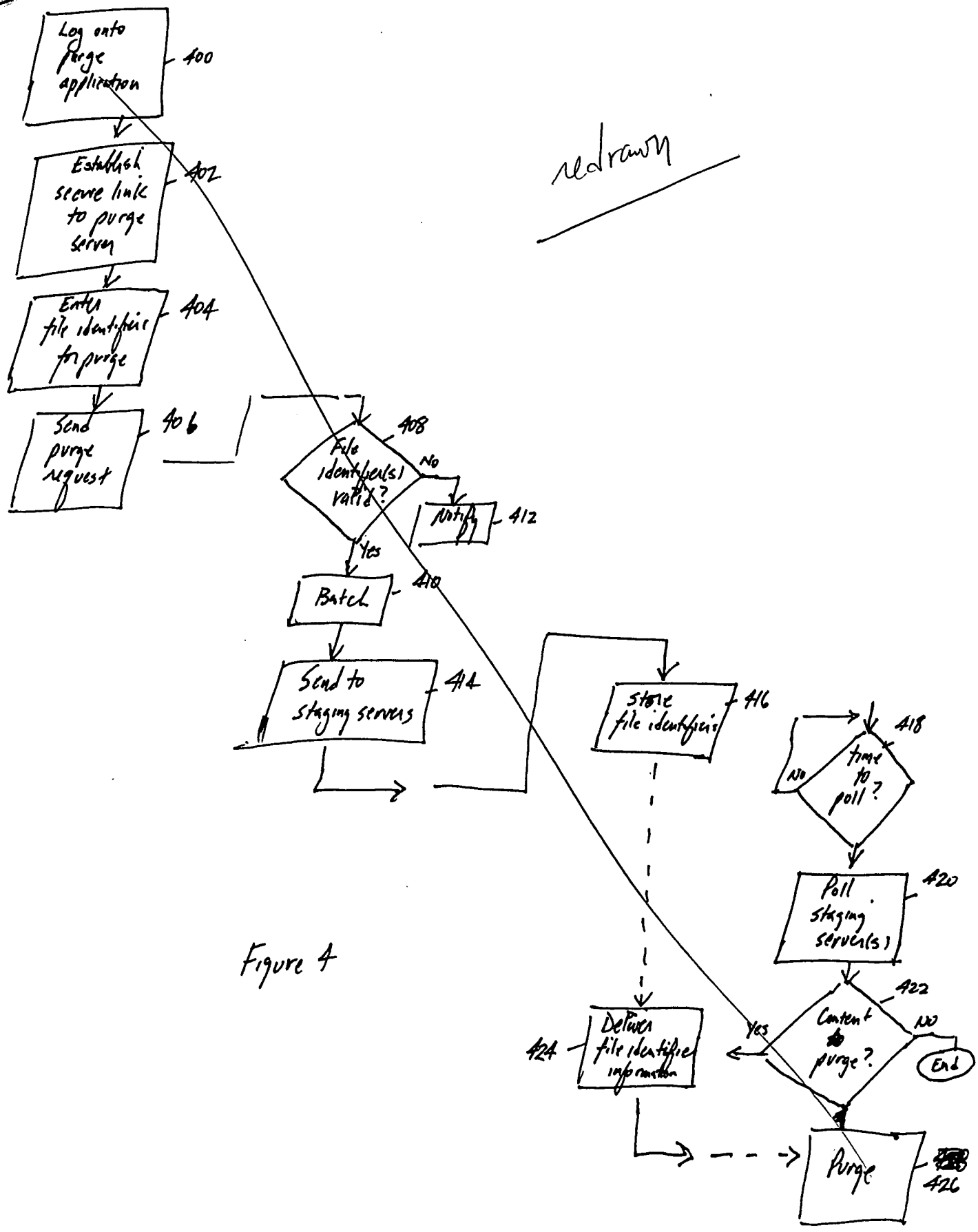



Figure 4

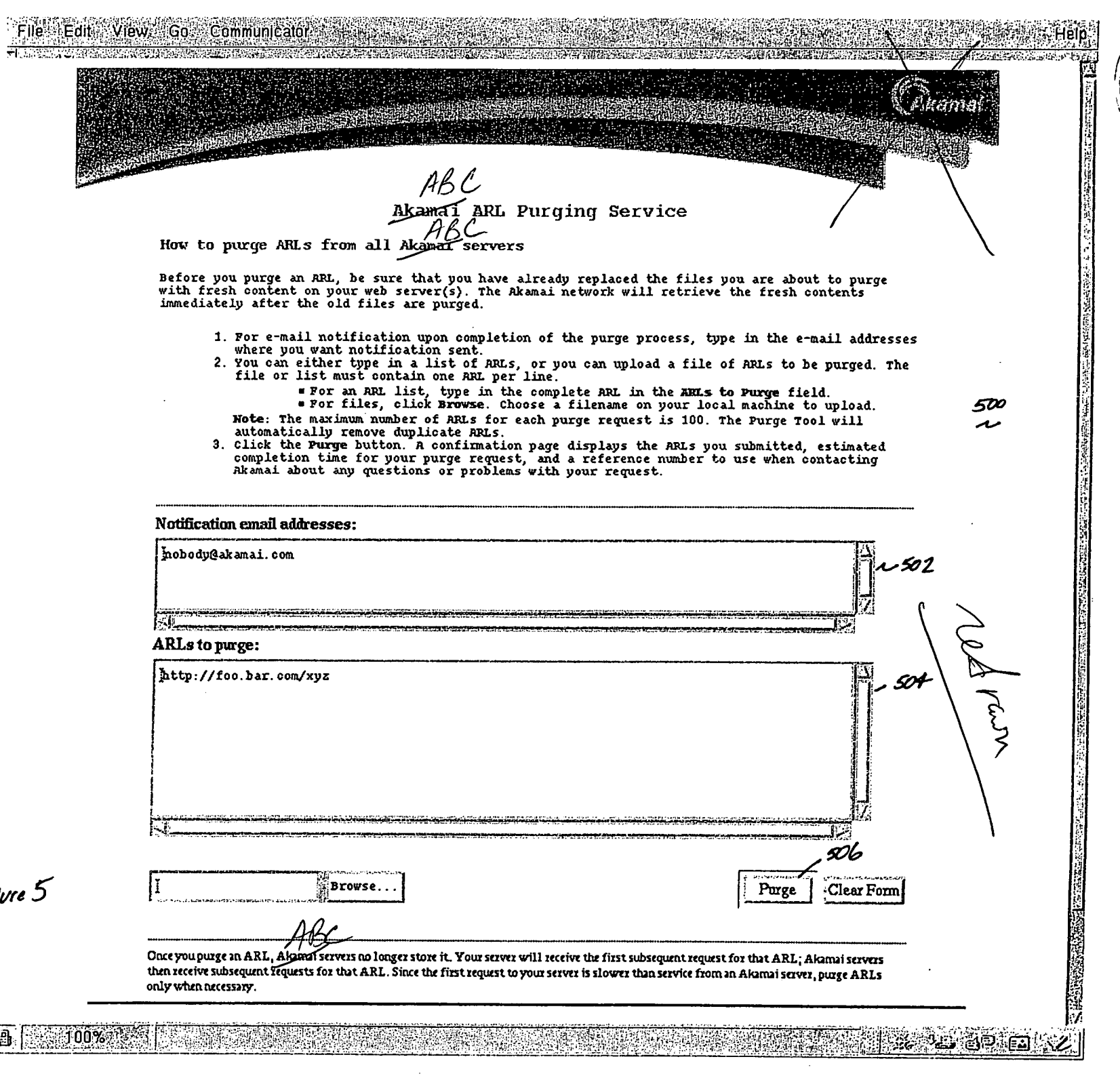



Figure 5



5/6

Annotated Sheet

Screen shot

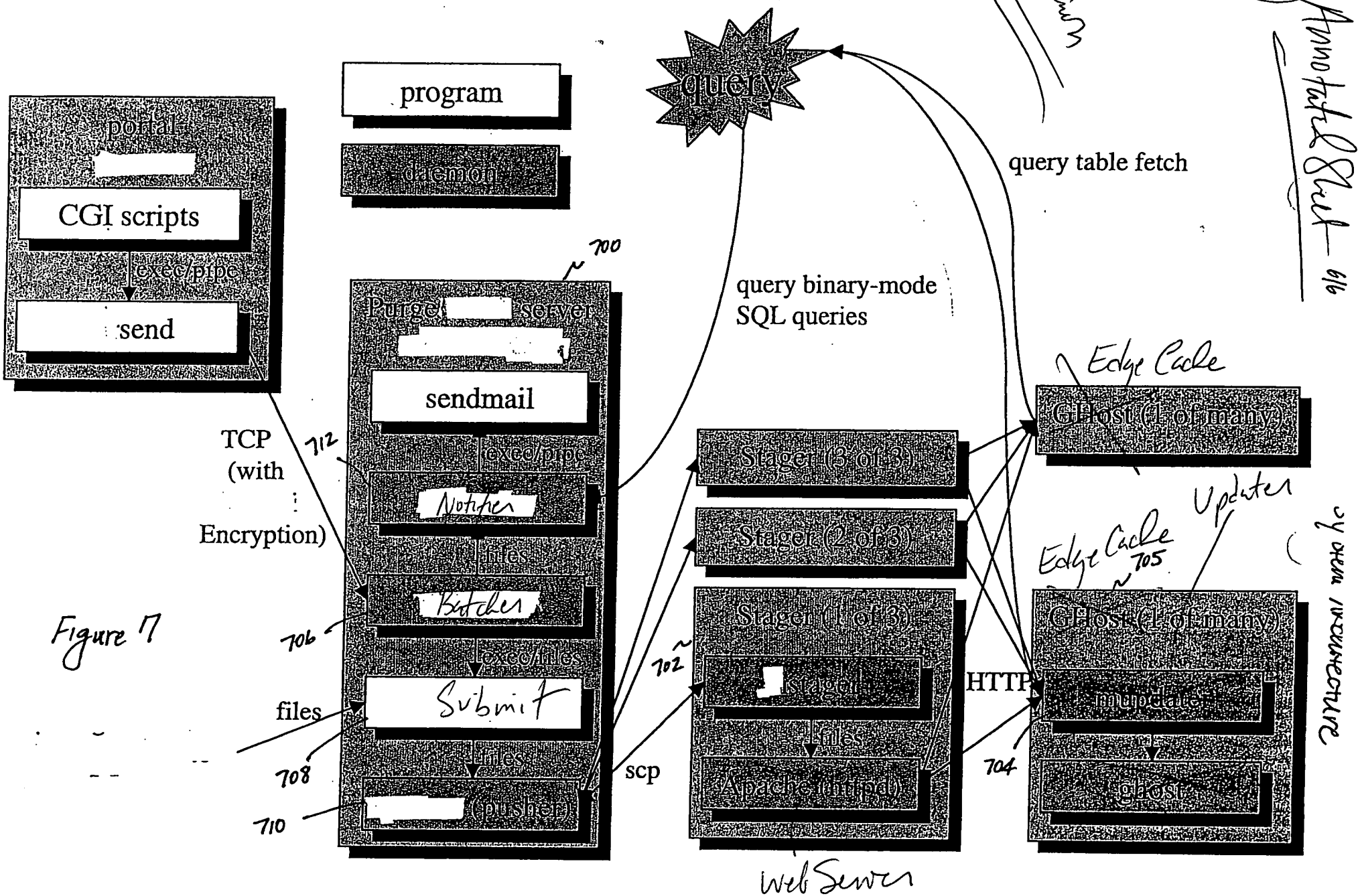



Figure 7